MULTIMEDIA UNIVERSITY

STUDENT ID NO

MULTIMEDIA UNIVERSITY

FINAL EXAMINATION

TRIMESTER 2, 2018/2019

TCP2451 – PROGRAMMING LANGUAGE TRANSLATION/ TCS 3311 – COMPILER DESIGN

(All Sections / Groups)

7 March 2019
2.30 PM– 4.30PM
(2 Hours)

INSTRUCTION TO STUDENT

1. Answer **ALL** questions.
2. This question paper has 4 printed pages excluding the front cover.
3. Please print all your answers in the answer booklet provided.

QUESTION 1 [25 marks]

a) Write a regular expression that can recognize the following token pattern. Define any assumption you have made (if any).

An operation consists of "OR" or "XOR". The operation ends with a ';' sign.

[4 marks]

b) Draw a NFA to recognize the regular expression in question 1(a).

[8 marks]

c) Using subset construction method, produce a transition table for a DFA derived from the NFA in question 1(b).

[8 marks]

d) Draw a DFA based on the transition table in question 1(c).

[5 marks]

Continued ....

QUESTION 2 [25 marks]

Consider the following context free grammar:

P	→ { L }
L	→ S L
L	→ ϵ
S	→ **id** = E;
E	→ NF
F	→ +NF
F	→ ϵ
N	→ (E) | **num**

P is the start symbol, and the terminals are **; { } id = + ( ) num.**

(a) Write the FIRST and FOLLOW sets for all the non-terminals in this grammar.

[12 marks]

(b) Draw the LL(1) parsing table for this grammar.

[6 marks]

(c) Based on the parsing table in Question 2(b), write a recursive procedure `parse_F` that parses this grammar by recursive descent. Your procedure may be written in Java, C++, Visual Basic or pseudocode. You may assume the existence of a global variable `nexttoken` that holds the next input token, a function `match()` that reads the next token into `nexttoken`, and a function parse_error that may be called if the input is not in the language generated by the grammar.

[7 marks]

Continued ....

QUESTION 3 [25 marks]

Given the following Java CUP specification file:

```
package myparser;

import java_cup.runtime.*;

/* REMOVED */

/*The grammar */
program ::= header body end;

header ::= OPENTAG TAGNAME CLOSETAG

body ::= LBRACE statement RBRACE;

statement ::= KEY ASSIGN VALUE;

end ::= OPENTAG TAGNAME CLOSETAG ENDTAG;
```

a) Complete the "Removed" commented section of this file so that it can be used to generate a parser.

[6 marks]

b) Give an example of a source program that confirms to this syntax. You must specify any assumption on the symbols used.

[6 marks]

c) Modify the grammar above so that the program body can contain COMMA separated *statement* recursively called.

[6 marks]

d) Write regular expression rules for the KEY and VALUE according to JLEX specification file format so that the scanner generated can be used together with the parser generated from this file. KEY contains only alphabet but VALUE can contain alphanumeric.

[7 marks]

Continued ....

QUESTION 4 [25 marks]

(a) Given the code statements:

```
        istore 1
Label_1:
        iload 1
        if_icmplt Label_2
        iconst 0
        ifeq Label_3
        iload 2
        goto Label_1
Label_2:
        iconst 1
Label_3:
        ifeq Label_2
        iload 3
        ireturn
```

Identify the basic blocks and draw the flow graph.

[9 marks]

(b) Given the following expression:

a * (b + c - d) + (b + c)

Explain how a Directed Acyclic Graph can be used to help to optimize the intermediate codes generated for this expression. Illustrate this by writing the set of optimized intermediate codes.

[8 marks]

(c) Given the following set of codes that could be generated for the statement "a := b + c", where R0 is a register and a, b, c are addressed memory locations of the variables.

Set 1	Set 2
MOV b, R0 ADD c, R0 MOV R0, a	MOV b, a ADD c, a

Calculate the instruction costs for each set of instructions and discuss the differences between the two sets of instructions.

[8 marks]
End of Page